UNIVERSIDADE FEDERAL DE ALAGOAS

Instituto de Computação
Ciência da Computação
Período 2018.1

Linguagem Sapphire

Lucas Ribeiro Raggi
Wagner da Silva Fontes

Analisador Sintático Buttom-up (SLR)

SLR foi o analisador sintático implementado para o reconhecimento da linguagem, a tabela encontra-se em: https://docs.google.com/spreadsheets/d/1C_0n_qhZaMu0-vSVwjKjsrC5yJ3R5RptqjK8UMk3-cY/edit?usp=sharing

Gramática

```
A = Sinicial
Sinicial = List_Func Decl_Main
Sinicial = Decl_Main

List_Func = List_Func Decl_Func
List_Func = Decl_Func

Decl_Func = FUNC Var_type ID BEGIN_PARAM Parameters END_PARAM
BEGIN_SCP Cmds END_SCP
Decl_Func = FUNC Var_type ID BEGIN_PARAM END_PARAM BEGIN_SCP Cmds
END_SCP


Decl_Main = FUNC Var_type MAIN BEGIN_PARAM END_PARAM BEGIN_SCP Cmds
END_SCP

Parameters = Parameters SEPARATOR Decl_Var
Parameters = Decl_Var

Decl_Var_r = Decl_Var_List
Decl_Var_r = Decl_Var
Decl_Var_r = Decl_Var_List OP_ATRIB E
Decl_Var_r = Decl_Var OP_ATRIB E

Decl_Var_List = Decl_Var_r SEPARATOR Decl_Var

Decl_Var = Var_type_r ID

Var_type_r = Var_type
Var_type_r = Var_arr_type

Var_arr_type = Var_type BEGIN_ARR E END_ARR
Var_arr_type = Var_type BEGIN_ARR END_ARR

Var_type = INT
Var_type = STR
Var_type = CHAR
Var_type = FLOAT
Var_type = BOOL
```

```
Var_type = VOID

E = E OP_ATRIB E_Or
E = E OP_CONCAT E_Or
E = E_Or

E_Or = E_Or OP_OR E_And
E_Or = E_And

E_And = E_And OP_AND E_Relat
E_And = E_Relat

E_Relat = E_Relat OP_REL_EQ E_Add
E_Relat = E_Relat OP_RELAT E_Add
E_Relat = E_Add

E_Add = E_Add OP_ADD E_Mult
E_Add = E_Mult

E_Mult = E_Mult OP_MULTI E_Exp
E_Mult = E_Exp

E_Exp = E_Exp OP_EXP E_Unneg
E_Exp = E_Unneg

E_Unneg = OP_UNNEG E_Neg
E_Unneg = E_Neg

E_Neg = OP_NEG Er
E_Neg = Er

Er = BEGIN_PARAM E END_PARAM
Er = Operand

Operand = Const
Operand = Call_Func
Operand = ID BEGIN_ARR E END_ARR
Operand = ID

Const = Numeric_Const
Const = CONST_STR
Const = CONST_BOOL
Const = CONST_CHAR
```

```
Numeric_Const = CONST_INT
Numeric_Const = CONST_FLT

Attrib = ID OP_ATRIB E
Attrib = Id_Arr OP_ATRIB E

Id_Arr = ID BEGIN_ARR E END_ARR

Call_Func = ID BEGIN_PARAM Call_Parameters END_PARAM
Call_Func = ID BEGIN_PARAM END_PARAM

Call_Parameters = Call_Parameters SEPARATOR E
Call_Parameters = E

Input = INS_INPUT BEGIN_PARAM Input_Param END_PARAM

Input_Param = Input_Param SEPARATOR Param_r
Input_Param = Param_r

Show = INS_SHOW BEGIN_PARAM E END_PARAM

Param_r = CONST_STR
Param_r = ID BEGIN_ARR E END_ARR
Param_r = ID BEGIN_ARR END_ARR
Param_r = ID

Cond = INS_IF E BEGIN_SCP Cmds END_SCP Elif_List
Cond = INS_IF E BEGIN_SCP Cmds END_SCP Else
Cond = INS_IF E BEGIN_SCP Cmds END_SCP

Elif_List = Elif_List INS_ELIF E BEGIN_SCP Cmds END_SCP
Elif_List = Elif_List INS_ELIF E BEGIN_SCP Cmds END_SCP Else
Elif_List = INS_ELIF E BEGIN_SCP Cmds END_SCP
Elif_List = INS_ELIF E BEGIN_SCP Cmds END_SCP Else

Else = INS_ELSE BEGIN_SCP Cmds END_SCP

Loop = INS_WHILE E BEGIN_SCP Cmds END_SCP
Loop = For Cmds END_SCP

For = INS_FOR Attrib SEPARATOR E SEPARATOR E BEGIN_SCP
For = INS_FOR Attrib SEPARATOR E SEPARATOR BEGIN_SCP
```

```
For = INS_FOR SEPARATOR E SEPARATOR E BEGIN_SCP
For = INS_FOR SEPARATOR E SEPARATOR BEGIN_SCP

Cmds = Cmd Cmds
Cmds = Cmd

Cmd = Decl_Var_r
Cmd = Rtrn
Cmd = Loop
Cmd = Cond
Cmd = Show
Cmd = Input
Cmd = Call_Func
Cmd = Attrib

Rtrn = INS_RETURN E
```

Hello World

```
|0001|  @ Código de teste
|0002|  func void main():
              [0002, 0001] (0000,       FUNC) {func}
              [0002, 0006] (0016,       VOID) {void}
          Var_type = VOID
              [0002, 0011] (0006,       MAIN) {main}
              [0002, 0015] (0002,BEGIN_PARAM) {(}
              [0002, 0016] (0003,  END_PARAM) {)}
              [0002, 0017] (0004,  BEGIN_SCP) {:}
|0003|  show("Hello world!")
              [0003, 0005] (0033,   INS_SHOW) {show}
              [0003, 0009] (0002,BEGIN_PARAM) {(}
              [0003, 0010] (0027,  CONST_STR) {"Hello world!"}
          Const = CONST_STR
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0003, 0024] (0003,  END_PARAM) {)}
|0004|  end
          Show = INS_SHOW BEGIN_PARAM E END_PARAM
          Cmd = Show
          Cmds = Cmd
              [0004, 0001] (0005,    END_SCP) {end}
          Decl_Main = FUNC Var_type MAIN BEGIN_PARAM END_PARAM
BEGIN_SCP Cmds END_SCP
          Sinicial = Decl_Main

 -------- ACC ---------
```

Shellsort

```
|0001|  @ ---------------
|0002|  @ Código de teste
|0003|  @ ---------------
|0004|
|0005|  func void shellsort(int[] arr, int n):
              [0005, 0001] (0000,       FUNC) {func}
              [0005, 0006] (0016,       VOID) {void}
            Var_type = VOID
              [0005, 0011] (0001,         ID) {shellsort}
              [0005, 0020] (0002,BEGIN_PARAM) {(}
              [0005, 0021] (0011,        INT) {int}
            Var_type = INT
              [0005, 0024] (0009,  BEGIN_ARR) {[}
              [0005, 0025] (0010,    END_ARR) {]}
            Var_arr_type = Var_type BEGIN_ARR END_ARR
            Var_type_r = Var_arr_type
              [0005, 0027] (0001,         ID) {arr}
            Decl_Var = Var_type_r ID
            Parameters = Decl_Var
              [0005, 0030] (0007,  SEPARATOR) {,}
              [0005, 0032] (0011,        INT) {int}
            Var_type = INT
            Var_type_r = Var_type
              [0005, 0036] (0001,         ID) {n}
            Decl_Var = Var_type_r ID
            Parameters = Parameters SEPARATOR Decl_Var
              [0005, 0037] (0003,  END_PARAM) {)}
              [0005, 0038] (0004,  BEGIN_SCP) {:}
|0006|
|0007|  int i, int j, int t, int temp
              [0007, 0005] (0011,        INT) {int}
            Var_type = INT
            Var_type_r = Var_type
              [0007, 0009] (0001,         ID) {i}
            Decl_Var = Var_type_r ID
            Decl_Var_r = Decl_Var
              [0007, 0010] (0007,  SEPARATOR) {,}
              [0007, 0012] (0011,        INT) {int}
            Var_type = INT
            Var_type_r = Var_type
```

```
            [0007, 0016] (0001,         ID) {j}
          Decl_Var = Var_type_r ID
          Decl_Var_List = Decl_Var_r SEPARATOR Decl_Var
          Decl_Var_r = Decl_Var_List
            [0007, 0017] (0007,  SEPARATOR) {,}
            [0007, 0019] (0011,        INT) {int}
          Var_type = INT
          Var_type_r = Var_type
            [0007, 0023] (0001,         ID) {t}
          Decl_Var = Var_type_r ID
          Decl_Var_List = Decl_Var_r SEPARATOR Decl_Var
          Decl_Var_r = Decl_Var_List
            [0007, 0024] (0007,  SEPARATOR) {,}
            [0007, 0026] (0011,        INT) {int}
          Var_type = INT
          Var_type_r = Var_type
            [0007, 0030] (0001,         ID) {temp}
|0008|  int i = (n/2)
          Decl_Var = Var_type_r ID
          Decl_Var_List = Decl_Var_r SEPARATOR Decl_Var
          Decl_Var_r = Decl_Var_List
          Cmd = Decl_Var_r
            [0008, 0005] (0011,        INT) {int}
          Var_type = INT
          Var_type_r = Var_type
            [0008, 0009] (0001,         ID) {i}
          Decl_Var = Var_type_r ID
            [0008, 0011] (0008,   OP_ATRIB) {=}
            [0008, 0013] (0002,BEGIN_PARAM) {(}
            [0008, 0014] (0001,         ID) {n}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
            [0008, 0015] (0023,   OP_MULTI) {/}
            [0008, 0016] (0030,  CONST_INT) {2}
          Numeric_Const = CONST_INT
          Const = Numeric_Const
          Operand = Const
          Er = Operand
          E_Neg = Er
```

```
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Mult OP_MULTI E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0008, 0017] (0003,  END_PARAM) {)}
|0009|  while i > 0:
          Er = BEGIN_PARAM E END_PARAM
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
          Decl_Var_r = Decl_Var OP_ATRIB E
          Cmd = Decl_Var_r
              [0009, 0005] (0037,  INS_WHILE) {while}
              [0009, 0011] (0001,         ID) {i}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
              [0009, 0013] (0021,   OP_RELAT) {>}
              [0009, 0015] (0030,  CONST_INT) {0}
          Numeric_Const = CONST_INT
          Const = Numeric_Const
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
```

```
            E_Relat = E_Relat OP_RELAT E_Add
            E_And = E_Relat
            E_Or = E_And
            E = E_Or
                [0009, 0016] (0004,  BEGIN_SCP) {:}
|0010|  for j = i, n - 1,:
                [0010, 0009] (0038,    INS_FOR) {for}
                [0010, 0013] (0001,         ID) {j}
                [0010, 0015] (0008,   OP_ATRIB) {=}
                [0010, 0017] (0001,         ID) {i}
            Operand = ID
            Er = Operand
            E_Neg = Er
            E_Unneg = E_Neg
            E_Exp = E_Unneg
            E_Mult = E_Exp
            E_Add = E_Mult
            E_Relat = E_Add
            E_And = E_Relat
            E_Or = E_And
            E = E_Or
            Attrib = ID OP_ATRIB E
                [0010, 0018] (0007,  SEPARATOR) {,}
                [0010, 0020] (0001,         ID) {n}
            Operand = ID
            Er = Operand
            E_Neg = Er
            E_Unneg = E_Neg
            E_Exp = E_Unneg
            E_Mult = E_Exp
            E_Add = E_Mult
                [0010, 0022] (0022,     OP_ADD) {-}
                [0010, 0024] (0030,  CONST_INT) {1}
            Numeric_Const = CONST_INT
            Const = Numeric_Const
            Operand = Const
            Er = Operand
            E_Neg = Er
            E_Unneg = E_Neg
            E_Exp = E_Unneg
            E_Mult = E_Exp
            E_Add = E_Add OP_ADD E_Mult
            E_Relat = E_Add
```

```
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0010, 0025] (0007,  SEPARATOR) {,}
              [0010, 0026] (0004,  BEGIN_SCP) {:}
|0011|  temp = arr[j]
          For = INS_FOR Attrib SEPARATOR E SEPARATOR BEGIN_SCP
              [0011, 0013] (0001,         ID) {temp}
              [0011, 0018] (0008,   OP_ATRIB) {=}
              [0011, 0020] (0001,         ID) {arr}
              [0011, 0023] (0009,  BEGIN_ARR) {[}
              [0011, 0024] (0001,         ID) {j}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0011, 0025] (0010,    END_ARR) {]}
|0012|  while t >= i and (arr[t - i] > temp):
          Operand = ID BEGIN_ARR E END_ARR
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
          Attrib = ID OP_ATRIB E
          Cmd = Attrib
              [0012, 0013] (0037,  INS_WHILE) {while}
              [0012, 0019] (0001,         ID) {t}
          Operand = ID
          Er = Operand
          E_Neg = Er
```

```
E_Unneg = E_Neg
E_Exp = E_Unneg
E_Mult = E_Exp
E_Add = E_Mult
E_Relat = E_Add
    [0012, 0021] (0021,   OP_RELAT) {>=}
    [0012, 0024] (0001,         ID) {i}
Operand = ID
Er = Operand
E_Neg = Er
E_Unneg = E_Neg
E_Exp = E_Unneg
E_Mult = E_Exp
E_Add = E_Mult
E_Relat = E_Relat OP_RELAT E_Add
E_And = E_Relat
    [0012, 0026] (0019,     OP_AND) {and}
    [0012, 0030] (0002,BEGIN_PARAM) {(}
    [0012, 0031] (0001,         ID) {arr}
    [0012, 0034] (0009,  BEGIN_ARR) {[}
    [0012, 0035] (0001,         ID) {t}
Operand = ID
Er = Operand
E_Neg = Er
E_Unneg = E_Neg
E_Exp = E_Unneg
E_Mult = E_Exp
E_Add = E_Mult
    [0012, 0037] (0022,     OP_ADD) {-}
    [0012, 0039] (0001,         ID) {i}
Operand = ID
Er = Operand
E_Neg = Er
E_Unneg = E_Neg
E_Exp = E_Unneg
E_Mult = E_Exp
E_Add = E_Add OP_ADD E_Mult
E_Relat = E_Add
E_And = E_Relat
E_Or = E_And
E = E_Or
    [0012, 0040] (0010,    END_ARR) {]}
Operand = ID BEGIN_ARR E END_ARR
```

```
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
              [0012, 0042] (0021,   OP_RELAT) {>}
              [0012, 0044] (0001,         ID) {temp}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Relat OP_RELAT E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0012, 0048] (0003,  END_PARAM) {)}
          Er = BEGIN_PARAM E END_PARAM
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_And OP_AND E_Relat
          E_Or = E_And
          E = E_Or
              [0012, 0049] (0004,  BEGIN_SCP) {:}
|0013|  arr[t] = arr[t - 1]
              [0013, 0017] (0001,         ID) {arr}
              [0013, 0020] (0009,  BEGIN_ARR) {[}
              [0013, 0021] (0001,         ID) {t}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
```

```
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0013, 0022] (0010,    END_ARR) {]}
          Id_Arr = ID BEGIN_ARR E END_ARR
              [0013, 0024] (0008,   OP_ATRIB) {=}
              [0013, 0026] (0001,         ID) {arr}
              [0013, 0029] (0009,  BEGIN_ARR) {[}
              [0013, 0030] (0001,         ID) {t}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
              [0013, 0032] (0022,     OP_ADD) {-}
              [0013, 0034] (0030,  CONST_INT) {1}
          Numeric_Const = CONST_INT
          Const = Numeric_Const
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Add OP_ADD E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0013, 0035] (0010,    END_ARR) {]}
|0014|  t = t - i
          Operand = ID BEGIN_ARR E END_ARR
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
```

```
          E_Or = E_And
          E = E_Or
          Attrib = Id_Arr OP_ATRIB E
          Cmd = Attrib
              [0014, 0017] (0001,          ID) {t}
              [0014, 0019] (0008,   OP_ATRIB) {=}
              [0014, 0021] (0001,          ID) {t}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
              [0014, 0023] (0022,     OP_ADD) {-}
              [0014, 0025] (0001,          ID) {i}
|0015|  end
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Add OP_ADD E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
          Attrib = ID OP_ATRIB E
          Cmd = Attrib
          Cmds = Cmd
          Cmds = Cmd Cmds
              [0015, 0013] (0005,    END_SCP) {end}
|0016|  arr[t] = temp
          Loop = INS_WHILE E BEGIN_SCP Cmds END_SCP
          Cmd = Loop
              [0016, 0013] (0001,          ID) {arr}
              [0016, 0016] (0009,  BEGIN_ARR) {[}
              [0016, 0017] (0001,          ID) {t}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
```

```
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0016, 0018] (0010,    END_ARR) {]}
          Id_Arr = ID BEGIN_ARR E END_ARR
              [0016, 0020] (0008,   OP_ATRIB) {=}
              [0016, 0022] (0001,         ID) {temp}
|0017|  end
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
          Attrib = Id_Arr OP_ATRIB E
          Cmd = Attrib
          Cmds = Cmd
          Cmds = Cmd Cmds
          Cmds = Cmd Cmds
              [0017, 0009] (0005,    END_SCP) {end}
|0018|  i = (i/2)
          Loop = For Cmds END_SCP
          Cmd = Loop
              [0018, 0009] (0001,         ID) {i}
              [0018, 0011] (0008,   OP_ATRIB) {=}
              [0018, 0013] (0002,BEGIN_PARAM) {(}
              [0018, 0014] (0001,         ID) {i}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
              [0018, 0015] (0023,   OP_MULTI) {/}
```

```
            [0018, 0016] (0030,  CONST_INT) {2}
          Numeric_Const = CONST_INT
          Const = Numeric_Const
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Mult OP_MULTI E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0018, 0017] (0003,  END_PARAM) {)}
|0019|  end
          Er = BEGIN_PARAM E END_PARAM
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
          Attrib = ID OP_ATRIB E
          Cmd = Attrib
          Cmds = Cmd
          Cmds = Cmd Cmds
              [0019, 0005] (0005,    END_SCP) {end}
|0020|  end
          Loop = INS_WHILE E BEGIN_SCP Cmds END_SCP
          Cmd = Loop
          Cmds = Cmd
          Cmds = Cmd Cmds
          Cmds = Cmd Cmds
              [0020, 0001] (0005,    END_SCP) {end}
|0021|
|0022|  func void main():
          Decl_Func = FUNC Var_type ID BEGIN_PARAM Parameters
END_PARAM BEGIN_SCP Cmds END_SCP
          List_Func = Decl_Func
```

```
            [0022, 0001] (0000,       FUNC) {func}
            [0022, 0006] (0016,       VOID) {void}
          Var_type = VOID
            [0022, 0011] (0006,       MAIN) {main}
            [0022, 0015] (0002,BEGIN_PARAM) {(}
            [0022, 0016] (0003,  END_PARAM) {)}
            [0022, 0017] (0004,  BEGIN_SCP) {:}
|0023|  int size
            [0023, 0005] (0011,        INT) {int}
          Var_type = INT
          Var_type_r = Var_type
            [0023, 0009] (0001,         ID) {size}
|0024|  int[300] arr
          Decl_Var = Var_type_r ID
          Decl_Var_r = Decl_Var
          Cmd = Decl_Var_r
            [0024, 0005] (0011,        INT) {int}
          Var_type = INT
            [0024, 0008] (0009,  BEGIN_ARR) {[}
            [0024, 0009] (0030,  CONST_INT) {300}
          Numeric_Const = CONST_INT
          Const = Numeric_Const
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
            [0024, 0012] (0010,    END_ARR) {]}
          Var_arr_type = Var_type BEGIN_ARR E END_ARR
          Var_type_r = Var_arr_type
            [0024, 0014] (0001,         ID) {arr}
|0025|  int i
          Decl_Var = Var_type_r ID
          Decl_Var_r = Decl_Var
          Cmd = Decl_Var_r
            [0025, 0005] (0011,        INT) {int}
          Var_type = INT
```

```
          Var_type_r = Var_type
              [0025, 0009] (0001,         ID) {i}
|0026|  show("Digite o tamanho da sequencia (limite de 300)")
          Decl_Var = Var_type_r ID
          Decl_Var_r = Decl_Var
          Cmd = Decl_Var_r
              [0026, 0005] (0033,   INS_SHOW) {show}
              [0026, 0009] (0002,BEGIN_PARAM) {(}
              [0026, 0010] (0027,  CONST_STR) {"Digite o tamanho da
sequencia (limite de 300)"}
          Const = CONST_STR
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0026, 0057] (0003,  END_PARAM) {)}
|0027|  input(size)
          Show = INS_SHOW BEGIN_PARAM E END_PARAM
          Cmd = Show
              [0027, 0005] (0032,  INS_INPUT) {input}
              [0027, 0010] (0002,BEGIN_PARAM) {(}
              [0027, 0011] (0001,         ID) {size}
          Param_r = ID
          Input_Param = Param_r
              [0027, 0015] (0003,  END_PARAM) {)}
|0028|  for i = 0, size - 1, :
          Input = INS_INPUT BEGIN_PARAM Input_Param END_PARAM
          Cmd = Input
              [0028, 0005] (0038,    INS_FOR) {for}
              [0028, 0009] (0001,         ID) {i}
              [0028, 0011] (0008,   OP_ATRIB) {=}
              [0028, 0013] (0030,  CONST_INT) {0}
          Numeric_Const = CONST_INT
          Const = Numeric_Const
          Operand = Const
          Er = Operand
```

```
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
          Attrib = ID OP_ATRIB E
              [0028, 0014] (0007,  SEPARATOR) {,}
              [0028, 0016] (0001,         ID) {size}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
              [0028, 0021] (0022,     OP_ADD) {-}
              [0028, 0023] (0030,  CONST_INT) {1}
          Numeric_Const = CONST_INT
          Const = Numeric_Const
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Add OP_ADD E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0028, 0024] (0007,  SEPARATOR) {,}
              [0028, 0026] (0004,  BEGIN_SCP) {:}
|0029|  input(arr[i])
          For = INS_FOR Attrib SEPARATOR E SEPARATOR BEGIN_SCP
              [0029, 0009] (0032,  INS_INPUT) {input}
              [0029, 0014] (0002,BEGIN_PARAM) {(}
              [0029, 0015] (0001,         ID) {arr}
              [0029, 0018] (0009,  BEGIN_ARR) {[}
              [0029, 0019] (0001,         ID) {i}
```

```
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0029, 0020] (0010,    END_ARR) {]}
          Param_r = ID BEGIN_ARR E END_ARR
          Input_Param = Param_r
              [0029, 0021] (0003,  END_PARAM) {)}
|0030|  end
          Input = INS_INPUT BEGIN_PARAM Input_Param END_PARAM
          Cmd = Input
          Cmds = Cmd
              [0030, 0005] (0005,    END_SCP) {end}
|0031|  show("array antes de ser ordenado")
          Loop = For Cmds END_SCP
          Cmd = Loop
              [0031, 0005] (0033,   INS_SHOW) {show}
              [0031, 0009] (0002,BEGIN_PARAM) {(}
              [0031, 0010] (0027,  CONST_STR) {"array antes de ser
ordenado"}
          Const = CONST_STR
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0031, 0039] (0003,  END_PARAM) {)}
|0032|  for i = 0, size - 2, :
          Show = INS_SHOW BEGIN_PARAM E END_PARAM
          Cmd = Show
```

```
    [0032, 0005] (0038,    INS_FOR) {for}
    [0032, 0009] (0001,         ID) {i}
    [0032, 0011] (0008,   OP_ATRIB) {=}
    [0032, 0013] (0030,  CONST_INT) {0}
Numeric_Const = CONST_INT
Const = Numeric_Const
Operand = Const
Er = Operand
E_Neg = Er
E_Unneg = E_Neg
E_Exp = E_Unneg
E_Mult = E_Exp
E_Add = E_Mult
E_Relat = E_Add
E_And = E_Relat
E_Or = E_And
E = E_Or
Attrib = ID OP_ATRIB E
    [0032, 0014] (0007,  SEPARATOR) {,}
    [0032, 0016] (0001,         ID) {size}
Operand = ID
Er = Operand
E_Neg = Er
E_Unneg = E_Neg
E_Exp = E_Unneg
E_Mult = E_Exp
E_Add = E_Mult
    [0032, 0021] (0022,     OP_ADD) {-}
    [0032, 0023] (0030,  CONST_INT) {2}
Numeric_Const = CONST_INT
Const = Numeric_Const
Operand = Const
Er = Operand
E_Neg = Er
E_Unneg = E_Neg
E_Exp = E_Unneg
E_Mult = E_Exp
E_Add = E_Add OP_ADD E_Mult
E_Relat = E_Add
E_And = E_Relat
E_Or = E_And
E = E_Or
    [0032, 0024] (0007,  SEPARATOR) {,}
```

```
            [0032, 0026] (0004,  BEGIN_SCP) {:}
|0033|  show(arr[i] & ", ")
          For = INS_FOR Attrib SEPARATOR E SEPARATOR BEGIN_SCP
              [0033, 0009] (0033,   INS_SHOW) {show}
              [0033, 0013] (0002,BEGIN_PARAM) {(}
              [0033, 0014] (0001,         ID) {arr}
              [0033, 0017] (0009,  BEGIN_ARR) {[}
              [0033, 0018] (0001,         ID) {i}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0033, 0019] (0010,    END_ARR) {]}
          Operand = ID BEGIN_ARR E END_ARR
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0033, 0021] (0017,  OP_CONCAT) {&}
              [0033, 0023] (0027,  CONST_STR) {", "}
          Const = CONST_STR
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
```

```
          E_Or = E_And
          E = E OP_CONCAT E_Or
              [0033, 0027] (0003,  END_PARAM) {)}
|0034|  end
          Show = INS_SHOW BEGIN_PARAM E END_PARAM
          Cmd = Show
          Cmds = Cmd
              [0034, 0005] (0005,    END_SCP) {end}
|0035|  shellsort(arr,size)
          Loop = For Cmds END_SCP
          Cmd = Loop
              [0035, 0005] (0001,         ID) {shellsort}
              [0035, 0014] (0002,BEGIN_PARAM) {(}
              [0035, 0015] (0001,         ID) {arr}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
          Call_Parameters = E
              [0035, 0018] (0007,  SEPARATOR) {,}
              [0035, 0019] (0001,         ID) {size}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
          Call_Parameters = Call_Parameters SEPARATOR E
              [0035, 0023] (0003,  END_PARAM) {)}
|0036|  show("array apos ser ordenado")
          Call_Func = ID BEGIN_PARAM Call_Parameters END_PARAM
```

```
          Cmd = Call_Func
              [0036, 0005] (0033,   INS_SHOW) {show}
              [0036, 0009] (0002,BEGIN_PARAM) {(}
              [0036, 0010] (0027,  CONST_STR) {"array apos ser
ordenado"}
          Const = CONST_STR
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0036, 0035] (0003,  END_PARAM) {)}
|0037|  for i = 0, size - 2, :
          Show = INS_SHOW BEGIN_PARAM E END_PARAM
          Cmd = Show
              [0037, 0005] (0038,    INS_FOR) {for}
              [0037, 0009] (0001,         ID) {i}
              [0037, 0011] (0008,   OP_ATRIB) {=}
              [0037, 0013] (0030,  CONST_INT) {0}
          Numeric_Const = CONST_INT
          Const = Numeric_Const
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
          Attrib = ID OP_ATRIB E
              [0037, 0014] (0007,  SEPARATOR) {,}
              [0037, 0016] (0001,         ID) {size}
          Operand = ID
          Er = Operand
```

```
            E_Neg = Er
            E_Unneg = E_Neg
            E_Exp = E_Unneg
            E_Mult = E_Exp
            E_Add = E_Mult
                [0037, 0021] (0022,     OP_ADD) {-}
                [0037, 0023] (0030,  CONST_INT) {2}
            Numeric_Const = CONST_INT
            Const = Numeric_Const
            Operand = Const
            Er = Operand
            E_Neg = Er
            E_Unneg = E_Neg
            E_Exp = E_Unneg
            E_Mult = E_Exp
            E_Add = E_Add OP_ADD E_Mult
            E_Relat = E_Add
            E_And = E_Relat
            E_Or = E_And
            E = E_Or
                [0037, 0024] (0007,  SEPARATOR) {,}
                [0037, 0026] (0004,  BEGIN_SCP) {:}
|0038|  show(arr[i] & ", ")
            For = INS_FOR Attrib SEPARATOR E SEPARATOR BEGIN_SCP
                [0038, 0009] (0033,   INS_SHOW) {show}
                [0038, 0013] (0002,BEGIN_PARAM) {(}
                [0038, 0014] (0001,         ID) {arr}
                [0038, 0017] (0009,  BEGIN_ARR) {[}
                [0038, 0018] (0001,         ID) {i}
            Operand = ID
            Er = Operand
            E_Neg = Er
            E_Unneg = E_Neg
            E_Exp = E_Unneg
            E_Mult = E_Exp
            E_Add = E_Mult
            E_Relat = E_Add
            E_And = E_Relat
            E_Or = E_And
            E = E_Or
                [0038, 0019] (0010,    END_ARR) {]}
            Operand = ID BEGIN_ARR E END_ARR
            Er = Operand
```

```
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0038, 0021] (0017,  OP_CONCAT) {&}
              [0038, 0023] (0027,  CONST_STR) {", "}
          Const = CONST_STR
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E OP_CONCAT E_Or
              [0038, 0027] (0003,  END_PARAM) {)}
|0039|  end
          Show = INS_SHOW BEGIN_PARAM E END_PARAM
          Cmd = Show
          Cmds = Cmd
              [0039, 0005] (0005,    END_SCP) {end}
|0040|  show(arr[size - 1])
          Loop = For Cmds END_SCP
          Cmd = Loop
              [0040, 0005] (0033,   INS_SHOW) {show}
              [0040, 0009] (0002,BEGIN_PARAM) {(}
              [0040, 0010] (0001,         ID) {arr}
              [0040, 0013] (0009,  BEGIN_ARR) {[}
              [0040, 0014] (0001,         ID) {size}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
```

```
          E_Add = E_Mult
              [0040, 0019] (0022,     OP_ADD) {-}
              [0040, 0021] (0030,  CONST_INT) {1}
          Numeric_Const = CONST_INT
          Const = Numeric_Const
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Add OP_ADD E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0040, 0022] (0010,    END_ARR) {]}
          Operand = ID BEGIN_ARR E END_ARR
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0040, 0023] (0003,  END_PARAM) {)}
|0041|  end
          Show = INS_SHOW BEGIN_PARAM E END_PARAM
          Cmd = Show
          Cmds = Cmd
          Cmds = Cmd Cmds
          Cmds = Cmd Cmds
          Cmds = Cmd Cmds
          Cmds = Cmd Cmds
          Cmds = Cmd Cmds
          Cmds = Cmd Cmds
          Cmds = Cmd Cmds
          Cmds = Cmd Cmds
          Cmds = Cmd Cmds
          Cmds = Cmd Cmds
```

```
          Cmds = Cmd Cmds
              [0041, 0001] (0005,    END_SCP) {end}
          Decl_Main = FUNC Var_type MAIN BEGIN_PARAM END_PARAM
BEGIN_SCP Cmds END_SCP
          Sinicial = List_Func Decl_Main

 -------- ACC ---------
```

Fibonacci

```
|0001|  @  -------------
|0002|  @ Código de teste
|0003|  @  -------------
|0004|
|0005|  func void fib(int n):
              [0005, 0001] (0000,       FUNC) {func}
              [0005, 0006] (0016,       VOID) {void}
          Var_type = VOID
              [0005, 0011] (0001,         ID) {fib}
              [0005, 0014] (0002,BEGIN_PARAM) {(}
              [0005, 0015] (0011,        INT) {int}
          Var_type = INT
          Var_type_r = Var_type
              [0005, 0019] (0001,         ID) {n}
          Decl_Var = Var_type_r ID
          Parameters = Decl_Var
              [0005, 0020] (0003,  END_PARAM) {)}
              [0005, 0021] (0004,  BEGIN_SCP) {:}
|0006|  int a = 0, int b = 1, int i = 0, int aux
              [0006, 0005] (0011,        INT) {int}
          Var_type = INT
          Var_type_r = Var_type
              [0006, 0009] (0001,         ID) {a}
          Decl_Var = Var_type_r ID
              [0006, 0011] (0008,   OP_ATRIB) {=}
              [0006, 0013] (0030,  CONST_INT) {0}
          Numeric_Const = CONST_INT
          Const = Numeric_Const
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
```

```
E = E_Or
Decl_Var_r = Decl_Var OP_ATRIB E
    [0006, 0014] (0007,  SEPARATOR) {,}
    [0006, 0016] (0011,         INT) {int}
Var_type = INT
Var_type_r = Var_type
    [0006, 0020] (0001,          ID) {b}
Decl_Var = Var_type_r ID
Decl_Var_List = Decl_Var_r SEPARATOR Decl_Var
    [0006, 0022] (0008,   OP_ATRIB) {=}
    [0006, 0024] (0030,  CONST_INT) {1}
Numeric_Const = CONST_INT
Const = Numeric_Const
Operand = Const
Er = Operand
E_Neg = Er
E_Unneg = E_Neg
E_Exp = E_Unneg
E_Mult = E_Exp
E_Add = E_Mult
E_Relat = E_Add
E_And = E_Relat
E_Or = E_And
E = E_Or
Decl_Var_r = Decl_Var_List OP_ATRIB E
    [0006, 0025] (0007,  SEPARATOR) {,}
    [0006, 0027] (0011,         INT) {int}
Var_type = INT
Var_type_r = Var_type
    [0006, 0031] (0001,          ID) {i}
Decl_Var = Var_type_r ID
Decl_Var_List = Decl_Var_r SEPARATOR Decl_Var
    [0006, 0033] (0008,   OP_ATRIB) {=}
    [0006, 0035] (0030,  CONST_INT) {0}
Numeric_Const = CONST_INT
Const = Numeric_Const
Operand = Const
Er = Operand
E_Neg = Er
E_Unneg = E_Neg
E_Exp = E_Unneg
E_Mult = E_Exp
E_Add = E_Mult
```

```
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
          Decl_Var_r = Decl_Var_List OP_ATRIB E
              [0006, 0036] (0007,  SEPARATOR) {,}
              [0006, 0038] (0011,        INT) {int}
          Var_type = INT
          Var_type_r = Var_type
              [0006, 0042] (0001,         ID) {aux}
|0007|  while (a + b) < n:
          Decl_Var = Var_type_r ID
          Decl_Var_List = Decl_Var_r SEPARATOR Decl_Var
          Decl_Var_r = Decl_Var_List
          Cmd = Decl_Var_r
              [0007, 0005] (0037,  INS_WHILE) {while}
              [0007, 0011] (0002,BEGIN_PARAM) {(}
              [0007, 0012] (0001,         ID) {a}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
              [0007, 0014] (0022,     OP_ADD) {+}
              [0007, 0016] (0001,         ID) {b}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Add OP_ADD E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0007, 0017] (0003,  END_PARAM) {)}
          Er = BEGIN_PARAM E END_PARAM
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
```

```
            E_Mult = E_Exp
            E_Add = E_Mult
            E_Relat = E_Add
                [0007, 0019] (0021,   OP_RELAT) {<}
                [0007, 0021] (0001,         ID) {n}
            Operand = ID
            Er = Operand
            E_Neg = Er
            E_Unneg = E_Neg
            E_Exp = E_Unneg
            E_Mult = E_Exp
            E_Add = E_Mult
            E_Relat = E_Relat OP_RELAT E_Add
            E_And = E_Relat
            E_Or = E_And
            E = E_Or
                [0007, 0022] (0004,  BEGIN_SCP) {:}
|0008|  if i > 0:
                [0008, 0009] (0034,     INS_IF) {if}
                [0008, 0012] (0001,         ID) {i}
            Operand = ID
            Er = Operand
            E_Neg = Er
            E_Unneg = E_Neg
            E_Exp = E_Unneg
            E_Mult = E_Exp
            E_Add = E_Mult
            E_Relat = E_Add
                [0008, 0014] (0021,   OP_RELAT) {>}
                [0008, 0016] (0030,  CONST_INT) {0}
            Numeric_Const = CONST_INT
            Const = Numeric_Const
            Operand = Const
            Er = Operand
            E_Neg = Er
            E_Unneg = E_Neg
            E_Exp = E_Unneg
            E_Mult = E_Exp
            E_Add = E_Mult
            E_Relat = E_Relat OP_RELAT E_Add
            E_And = E_Relat
            E_Or = E_And
            E = E_Or
```

```
              [0008, 0017] (0004,  BEGIN_SCP) {:}
|0009|  show(",")
              [0009, 0013] (0033,   INS_SHOW) {show}
              [0009, 0017] (0002,BEGIN_PARAM) {(}
              [0009, 0018] (0027,  CONST_STR) {","}
            Const = CONST_STR
            Operand = Const
            Er = Operand
            E_Neg = Er
            E_Unneg = E_Neg
            E_Exp = E_Unneg
            E_Mult = E_Exp
            E_Add = E_Mult
            E_Relat = E_Add
            E_And = E_Relat
            E_Or = E_And
            E = E_Or
              [0009, 0021] (0003,  END_PARAM) {)}
|0010|  end
            Show = INS_SHOW BEGIN_PARAM E END_PARAM
            Cmd = Show
            Cmds = Cmd
              [0010, 0009] (0005,    END_SCP) {end}
|0011|  if i == 1:
            Cond = INS_IF E BEGIN_SCP Cmds END_SCP
            Cmd = Cond
              [0011, 0009] (0034,     INS_IF) {if}
              [0011, 0012] (0001,         ID) {i}
            Operand = ID
            Er = Operand
            E_Neg = Er
            E_Unneg = E_Neg
            E_Exp = E_Unneg
            E_Mult = E_Exp
            E_Add = E_Mult
            E_Relat = E_Add
              [0011, 0014] (0020,  OP_REL_EQ) {==}
              [0011, 0017] (0030,  CONST_INT) {1}
            Numeric_Const = CONST_INT
            Const = Numeric_Const
            Operand = Const
            Er = Operand
            E_Neg = Er
```

```
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Relat OP_REL_EQ E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0011, 0018] (0004,  BEGIN_SCP) {:}
|0012|  show("0")
              [0012, 0013] (0033,   INS_SHOW) {show}
              [0012, 0017] (0002,BEGIN_PARAM) {(}
              [0012, 0018] (0027,  CONST_STR) {"0"}
          Const = CONST_STR
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0012, 0021] (0003,  END_PARAM) {)}
|0013|  end
          Show = INS_SHOW BEGIN_PARAM E END_PARAM
          Cmd = Show
          Cmds = Cmd
              [0013, 0009] (0005,    END_SCP) {end}
|0014|  if i == 1:
          Cond = INS_IF E BEGIN_SCP Cmds END_SCP
          Cmd = Cond
              [0014, 0009] (0034,     INS_IF) {if}
              [0014, 0012] (0001,         ID) {i}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
```

```
          E_Relat = E_Add
              [0014, 0014] (0020,  OP_REL_EQ) {==}
              [0014, 0017] (0030,  CONST_INT) {1}
          Numeric_Const = CONST_INT
          Const = Numeric_Const
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Relat OP_REL_EQ E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0014, 0018] (0004,  BEGIN_SCP) {:}
|0015|  show("1")
              [0015, 0013] (0033,   INS_SHOW) {show}
              [0015, 0017] (0002,BEGIN_PARAM) {(}
              [0015, 0018] (0027,  CONST_STR) {"1"}
          Const = CONST_STR
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
              [0015, 0021] (0003,  END_PARAM) {)}
|0016|  end
          Show = INS_SHOW BEGIN_PARAM E END_PARAM
          Cmd = Show
          Cmds = Cmd
              [0016, 0009] (0005,    END_SCP) {end}
|0017|  else:
              [0017, 0009] (0036,   INS_ELSE) {else}
              [0017, 0013] (0004,  BEGIN_SCP) {:}
|0018|  aux = a + b
```

```
            [0018, 0013] (0001,          ID) {aux}
            [0018, 0017] (0008,   OP_ATRIB) {=}
            [0018, 0019] (0001,          ID) {a}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
            [0018, 0021] (0022,     OP_ADD) {+}
            [0018, 0023] (0001,          ID) {b}
|0019|  show(aux)
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Add OP_ADD E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
          Attrib = ID OP_ATRIB E
          Cmd = Attrib
            [0019, 0013] (0033,   INS_SHOW) {show}
            [0019, 0017] (0002,BEGIN_PARAM) {(}
            [0019, 0018] (0001,          ID) {aux}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
            [0019, 0021] (0003,  END_PARAM) {)}
|0020|  a = b
          Show = INS_SHOW BEGIN_PARAM E END_PARAM
```

```
          Cmd = Show
              [0020, 0013] (0001,          ID) {a}
              [0020, 0015] (0008,   OP_ATRIB) {=}
              [0020, 0017] (0001,          ID) {b}
|0021|  b = a + b
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
          Attrib = ID OP_ATRIB E
          Cmd = Attrib
              [0021, 0013] (0001,          ID) {b}
              [0021, 0015] (0008,   OP_ATRIB) {=}
              [0021, 0017] (0001,          ID) {a}
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
              [0021, 0019] (0022,     OP_ADD) {+}
              [0021, 0021] (0001,          ID) {b}
|0022|  end
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Add OP_ADD E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
          Attrib = ID OP_ATRIB E
```

```
          Cmd = Attrib
          Cmds = Cmd
          Cmds = Cmd Cmds
          Cmds = Cmd Cmds
          Cmds = Cmd Cmds
              [0022, 0009] (0005,    END_SCP) {end}
|0023|  i = 1 + i
          Else = INS_ELSE BEGIN_SCP Cmds END_SCP
          Cond = INS_IF E BEGIN_SCP Cmds END_SCP Else
          Cmd = Cond
              [0023, 0005] (0001,         ID) {i}
              [0023, 0007] (0008,   OP_ATRIB) {=}
              [0023, 0009] (0030,  CONST_INT) {1}
          Numeric_Const = CONST_INT
          Const = Numeric_Const
          Operand = Const
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
              [0023, 0011] (0022,     OP_ADD) {+}
              [0023, 0013] (0001,         ID) {i}
|0024|  end
          Operand = ID
          Er = Operand
          E_Neg = Er
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Add OP_ADD E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
          Attrib = ID OP_ATRIB E
          Cmd = Attrib
          Cmds = Cmd
          Cmds = Cmd Cmds
          Cmds = Cmd Cmds
          Cmds = Cmd Cmds
              [0024, 0005] (0005,    END_SCP) {end}
```

```
|0025|  end
          Loop = INS_WHILE E BEGIN_SCP Cmds END_SCP
          Cmd = Loop
          Cmds = Cmd
          Cmds = Cmd Cmds
              [0025, 0001] (0005,    END_SCP) {end}
|0026|
|0027|
|0028|  func void main():
          Decl_Func = FUNC Var_type ID BEGIN_PARAM Parameters
END_PARAM BEGIN_SCP Cmds END_SCP
          List_Func = Decl_Func
              [0028, 0001] (0000,       FUNC) {func}
              [0028, 0006] (0016,       VOID) {void}
          Var_type = VOID
              [0028, 0011] (0006,       MAIN) {main}
              [0028, 0015] (0002,BEGIN_PARAM) {(}
              [0028, 0016] (0003,  END_PARAM) {)}
              [0028, 0017] (0004,  BEGIN_SCP) {:}
|0029|  int n
              [0029, 0005] (0011,        INT) {int}
          Var_type = INT
          Var_type_r = Var_type
              [0029, 0009] (0001,         ID) {n}
|0030|  input(n)
          Decl_Var = Var_type_r ID
          Decl_Var_r = Decl_Var
          Cmd = Decl_Var_r
              [0030, 0005] (0032,  INS_INPUT) {input}
              [0030, 0010] (0002,BEGIN_PARAM) {(}
              [0030, 0011] (0001,         ID) {n}
          Param_r = ID
          Input_Param = Param_r
              [0030, 0012] (0003,  END_PARAM) {)}
|0031|  fib(n)
          Input = INS_INPUT BEGIN_PARAM Input_Param END_PARAM
          Cmd = Input
              [0031, 0005] (0001,         ID) {fib}
              [0031, 0008] (0002,BEGIN_PARAM) {(}
              [0031, 0009] (0001,         ID) {n}
          Operand = ID
          Er = Operand
          E_Neg = Er
```

```
          E_Unneg = E_Neg
          E_Exp = E_Unneg
          E_Mult = E_Exp
          E_Add = E_Mult
          E_Relat = E_Add
          E_And = E_Relat
          E_Or = E_And
          E = E_Or
          Call_Parameters = E
              [0031, 0010] (0003,  END_PARAM) {)}
|0032|  end
          Call_Func = ID BEGIN_PARAM Call_Parameters END_PARAM
          Cmd = Call_Func
          Cmds = Cmd
          Cmds = Cmd Cmds
          Cmds = Cmd Cmds
              [0032, 0001] (0005,    END_SCP) {end}
          Decl_Main = FUNC Var_type MAIN BEGIN_PARAM END_PARAM
BEGIN_SCP Cmds END_SCP
          Sinicial = List_Func Decl_Main

 -------- ACC ---------
```